Delay Computation for Real-Time Synchronization of Speech and its Converted Text

Hamida Qunber Ali
Department of Computer Science
Iqra University
Karachi, Pakistan
hameeda_ali@yahoo.co.uk

Jameel Ahmed
Department of Electronic and Computer Engineering
NFC IET
Multan, Pakistan
jameel@nfciet.edu.pk

Mohammed Yakoob Siyal
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore
eyakoob@ntu.edu.sg

Abstract— **Transmission of real-time text data integrated with other multimedia applications such as audio and video has raised the issues of compatibility and synchronization among these applications since a stringent quality of service (QoS) guarantee is especially critical for real-time traffic. In order to meet the real-time properties, text must be produced efficiently to integrate its transmission with other multimedia applications. Literature survey shows that the text for the real-time transmission can be produced by different input sources such as from handwriting recognition, voice recognition or it can be entered by human users from a keyboard or any other input method [1]. This paper presents an efficient way of producing text which to the best of our knowledge has not been previously explored. We propose to generate text from the recognition of the real-time voice in the source machine. We calculate the delay in the speech-recognition or speech-to-text conversion. Based on these statistics we suggest a buffer size to store the voice data until its respective text is generated. This enables us to transmit both voice and its converted text synchronously. We find, and show it graphically, that this delay is almost negligible and there is almost no queue formation in the buffer. Hence both of the applications can be transmitted instantly that is as they are available. This research is a reasonable advancement in the subject area**.

Keywords—component, formatting, style, styling, insert

I. INTRODUCTION

Internet utilization has continued to grow exponentially with the unbound advancement in different multimedia applications such as audio, video and text. In the present phase there is a movement to integrate different media applications for example audio, video conferencing. Likewise, combining speech with its real-time converted text has given rise to several useful applications such as distance-education for deaf people, multi-lingual learning and teaching and textual display of voice conversation on the mobile screen etc.

In this work we have generated real-time text from the real-time speech using the speech-recognition process. Although there are different methods of generating real-time text e.g. from handwriting recognition or from keyboard entry, however, these methods are prone to human error and delay and can not meet the stringent QoS requirements of real-time transmission [1]. Among these techniques speech recognition is the most appropriate to generate text for real-time transmission. Most commercial companies claim that recognition software can achieve between 98% to 99% accuracy (i.e getting one to two words out of one hundred wrong) if operated under optimal conditions [2].

Section II gives an overview of the present speech recognition technology. In section III we describe our work methodology, section IV presents the calculation of buffer size and in section V we present the results evaluating the performance of our proposed work.

II. SPEECH RECOGNITION

Speech recognition is the process of converting a speech signal to a sequence of words, by means of an algorithm implemented as a computer program. Speech recognition applications that have emerged over the last years include voice dialing (e.g. *Call home*), call routing (e.g., *I would like to make a collect call*), simple data entry (e.g., entering a credit card number), and preparation of structured documents (e.g., a radiology report). The performance of speech recognition systems is usually specified in terms of accuracy and speed. Accuracy is measured with the *word error rate*, whereas speed is measured with the real time factor[1]. The growing advancement and high accuracy rate of speech recognition technology has made it the future technology.

For some time now researchers have been working to achieve least word error rate (WER %) by combining the baseline recognizers which might produce a final system more accurate than either of the constituents alone. I. Lee

Hetherington experimented merging standard hidden Markov models (HMMs) with landmark models and showed that combined acoustic models achieve an improved WER than either baseline models alone [3].

Hugo Meinedo developed hybrid systems that combined HMM and multilayer perceptions (MLPs) where was possible to obtain relative improvements on WER larger than 20% for a large vocabulary speaker independent continuous speech recognition task [4].

[1] The real time factor (RTF) is a common metric for measuring the speed of an automatic speech recognition system.

If it takes time *P* to process an input of length *I* the real time factor is defined as

RTF = P / *l*

If for example a recording of length 2 hours is processed in 8 hours computation time, the real time factor is 4. When the real time factor is 1, the processing is done in *real time*. It is a hardware dependent value.

III. PROPOSED SOLUTION

The system description, proposed in this paper, is shown in Fig.1. The target service is to transmit real-time speech and its converted text synchronously from the source machine. The sound card in the source machine modulates the acoustic signal coming from microphone into voice data using 16-bit Pulse Code Modulation (PCM) which gives a 128 kilobit per second (kbps) digital signal [5].

Microphone input is typically real-time audio object. This means that the audio object is designed to support audio buffering and dynamic state manipulation (e.g. stop->play->pause->play->stop) to handle delays and latency in the audio source and/or the SR engine's processing [6]. In our scheme we propose to send a copy of the voice data, coming from the A/D converter box, for speech recognition process to generate real-time text. We buffer the original voice data for the conversion time as shown in the Fig.1.

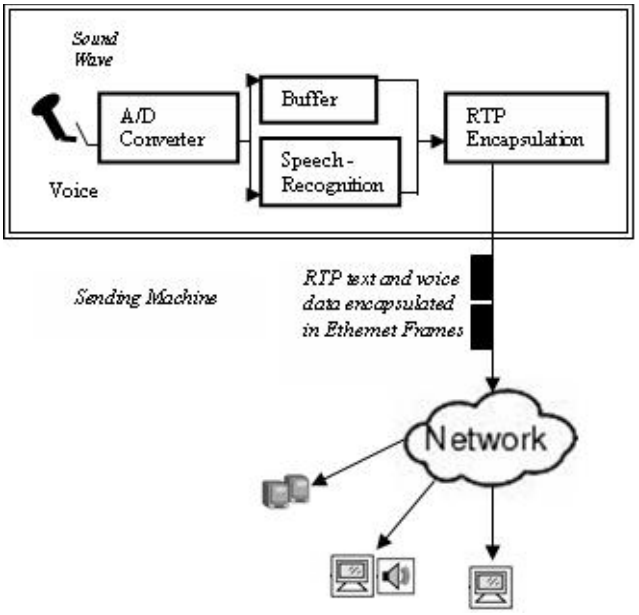

Figure 1. Real-Time Speech and Text Transmission. A copy of the voice data is sent for speech-recognition to convert it into text data while the original voice data is buffered until the respective text is available. We calculate the required buffer size.

We estimate the speech recognition and conversion delay for each byte of voice data by using our software program and determine how long each byte needs to be buffered in order to synchronize its transmission with its respective text data using real-time transport protocol (RTP).

Our software program, developed in Microsoft Visual Basic has a predefined set of words. The speech recognition embedded in the program can only recognize those words. The purpose of this program is to find the average speech-to-text conversion delay per word. We further calculate the average human speaking rate (words per second). These statistics assist us to calculate an average buffer size to store the voice data so long as its respective text data is generated by the speech recognition process. This enables us to transmit both the real-time services synchronously.

IV. BUFFER SIZE CALCULATION

In this section our target is to estimate how long the voice data should be buffered. For this we first calculate the time it takes to generate voice data for real-time transmission using RTP which provides transport services for the transmission of real-time video, voice and text data on top of UDP. Second we calculate the delay in the speech-to-text conversion process. Based on these statistics we propose a buffer size to store the voice data and transmit it with its respective text data when it is available.

A. Delay Computation for generating Real-Time Voice Data

Following are the salient points we used in our work:

- Analog voice is sampled at 8000 samples per second.
- 16-bit PCM is used to encode analog voice for digital transmission.
- It is assumed that Ethernet provides layer two framing for voice and text transmission.
- There is only one type of data being served (no priority queuing).
- The serve order is FIFO (first in-first out).
- Single server queue (one instance of RTP provides packetization).
- Constant serving rate.
- Fixed serving time (RTP packetization rate is fixed).

1) Speech Encoding Delay: A 16-bit PCM is an efficient encoding scheme for analog voice signal. Sampling voice 8,000 times in one second gives a 128 kilo bits per second or 16 kilo bytes per second (kBps) digital signal.
Encoding delay = 1/16000 = 62.5 μs/byte

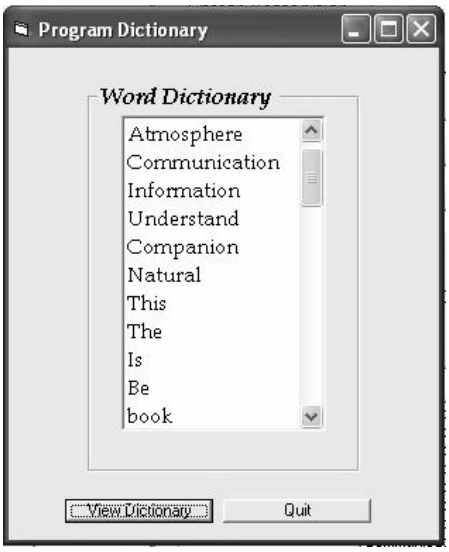

Figure 2. Program Dictionary

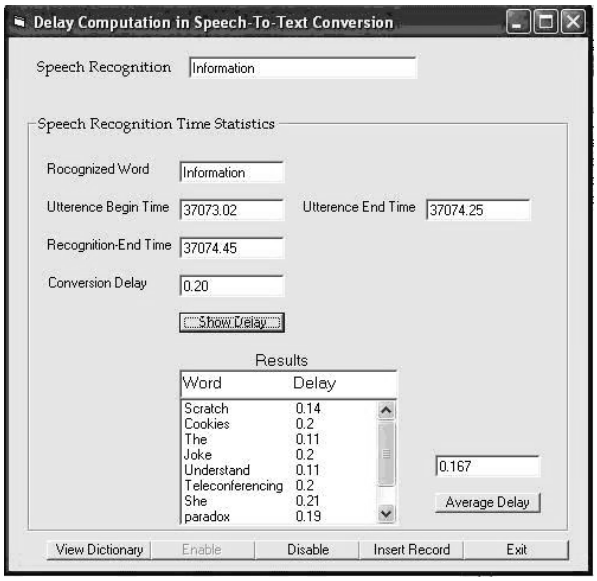

Figure 3. Form Layout of the Program

2) Voice Data Packetization Delay: The RTP data packets contain no length field or other delineation; therefore it relies on the underlying protocol(s) to provide a length indication [7]. Since each upper layer protocol PDU is encapsulated into the payload of its successive underlying protocol, the RTP data packet will ultimately be encapsulated in the Ethernet frame. We, therefore, find the mean length of RTP data packets by subtracting the IP, UDP and RTP header lengths from the Ethernet frame's maximum (1500 bytes) and minimum payload size (46 bytes) and then average these two extreme payload sizes.

B. Delay Computation for Speech-to-Text Conversion

In this section, we find the delay in the conversion of the real-time speech into text for a single word and ultimately for a single byte. This is carried out in the following two steps.

1) Average Human Speaking Rate: We choose ten volunteers of different ages and sex to read some randomly chosen text in his/her normal speaking tone for one minute and note his/her speaking rate. Each volunteer performs this experiment ten times and finds his/her average speaking rate. For further accuracy we average the speaking rate from all volunteers. Some of the sample readings are shown in Table1. Average human speaking rate is 119.94 or 120 words per minute which is equal to 2 words per second.

TABLE 1. AVERAGE HUMAN SPEAKING RATE

	Users				
	A	**B**	**C**	**D**	**E**
Number of Words Spoken in One Minute	122	118	119	119	123
	120	117	120	119	122
	122	118	122	117	124
	120	119	120	118	122
	120	117	119	120	122
	120	118	121	118	119
	123	118	121	119	121
	122	119	118	120	118
	120	119	121	121	120
	120	117	118	120	120
Average	**120.9**	**118**	**119.9**	**119.1**	**121.1**

2) Speech-to-Text Conversion Delay: In this section we find the delay in the conversion of speech into text. We developed a software program in Visual Basic in which we integrated Microsoft Speech Recognition API and its Direct Speech Recognition object to add speech recognition features. The program has a dictionary of fifty words which it can recognize as shown in Fig.2. If an out of vocabulary (OOV) word is spoken the program informs the user by typing "No Match".

The Enable button activates the speech engine. The program calls the system timer at different events such as utterance begin, utterance end and recognition end and calculates the delay in the conversion of spoken word into text.

V. RESULTS

A. Delay in Generating Real-Time Voice Data

We find the mean RTP payload size by subtracting the IP, UDP and RTP header lengths from the maximum and minimum payload length of Ethernet frame.

Maximum Length of a RTP Payload = 1500 – [20(IP) + 8(UDP) + 12(RTP)] = 1460 bytes

Minimum Length of a RTP Payload = 46 – [20(IP) + 8(UDP) + 12(RTP)] = 6 bytes

Mean Length of RTP Payload = (1460+ 6) / 2 = 733 bytes

RTP packetization interval for audio data = 20 ms [1]

In other words RTP encapsulate 733 Bytes of data every 20ms.

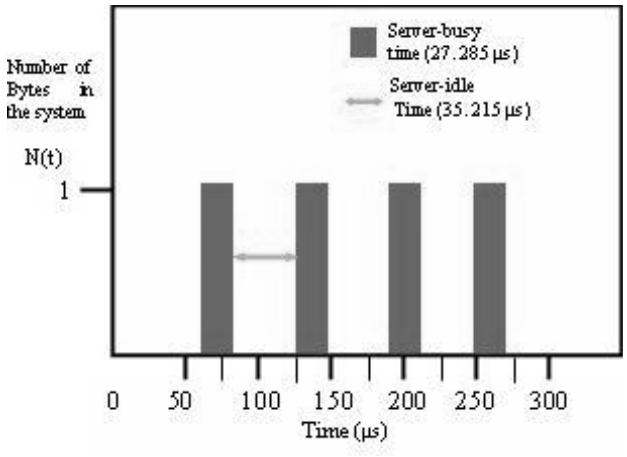

Figure 4.

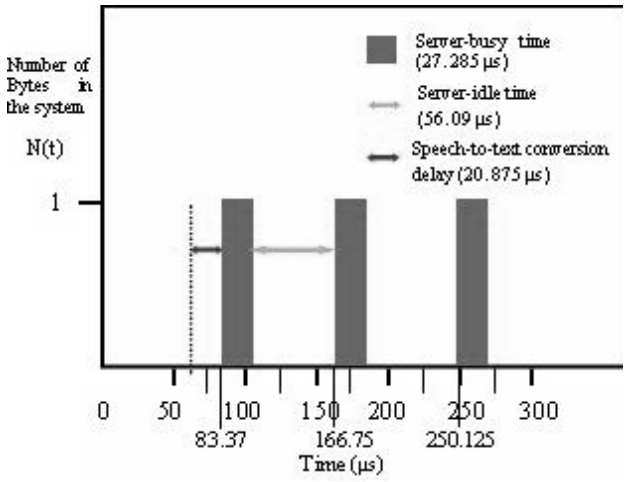

Figure 5. Server Idle Time after Speech-to-Text Conversion

In one second RTP encapsulates (733/20) × 1000 = 36,650bytes

As the rate of the encoded voice data is 16000 bytes/s

Data arrival rate = 16,000 bytes/s

RTP service rate = 36,650 bytes/s

Since the Service Rate > Arrival Rate

This implies that there is no queue formation in the buffer when only voice is transmitted using RTP.

Mean service time for one byte = 1/ 36650 = 27.285 µs

Server (RTP providing Real-Time transport) idle time =Encoding Delay – Service Time = 62.5 – 27.285 = 35.215µs/byte

B. Delay in Speech-to-Text Conversion

We use our software program to find the speech-to-text conversion delay by uttering any word in the mike from our program dictionary. The program calls the system timer on different events and calculates the average conversion delay for the uttered word. We got an average conversion delay of 0.167 seconds per word.

Average speaking rate (shown in Table 1) = 2 words/s

Conversion delay for two words (16000 bytes) = 2 x 0.167 = 0.334 s

Conversion delay for a single byte =0.334/16000= 20.875 µs

Server idle time after speech conversion = Server idle time before conversion + Speech Conversion Delay = 35.215 + 20.875 = 56.09 µs

C. Synchronization of Speech and Text Transmission

The calculations above show the conversion delay for a single word which is 0.167 seconds or 20.875 µseconds for a single byte. This implies that each byte of voice data should be buffered for 20.875 µseconds until a copy of it is converted into the respective text. This scheme gives a nominal delay to generate text for real-time transmission and can be efficiently used to synchronize its transmission with the real-time speech.

D. Queuing Analysis

We, further, perform queuing analysis to find the size of the buffer for storing the voice data.

Total delay after speech conversion = Encoding Delay + Speech-to-Text Conversion Delay = 62.5 + 20.875 = 83.375µs/ byte

Or Number of bytes arriving for RTP encapsulation (after conversion) in 82.5µs = 1

Number of bytes arriving in one second=1000000 / 83.375 = 11994.0029 bytes

Arrival rate (after conversion) $\lambda \approx 11995$ bytes/s

The service rate $\mu = 36650$ bytes/s

Average traffic intensity $\rho = \lambda/\mu = 11995 / 36650 = 0.327$

As $\rho < 1 \Rightarrow$ the system is stable in the long run.

Average buffer size $N_Q = \rho 2 / (1 - \rho) = 0.327^2 / (1 – 0.327) = 0.159$

$N_Q \approx 1$ byte

Our results show that there is only one byte in the buffer at any time which indicates that there is almost no queue formation in our proposed system. Hence generating text data for real-time transmission from the recognition of the speech is an efficient and more accurate way as compared to other conventional methods.

E. At the Receiver

At the receiving end, a reverse process of the sending side will be performed. The RTP audio and text packets will be decapsulated and decoded into raw audio and text stream. As the audio is played out after every 20 ms, the text data will be buffered for this time interval which will allow synchronization of both media at the receiving end.

The Internet, like other packet networks, occasionally loses and reorders packets and delays them by variable amounts of

time. To cope with these impairments, the RTP header contains timing information and a sequence number that allows the receivers to reconstruct the timing produced by the source, so that, chunks of audio are contiguously played out by the speaker every 20 ms. This timing reconstruction is also performed separately for text packets in the conference. The sequence number can also be used by the receiver to estimate how many packets are being lost.

VI. CONCLUSION

We presented a way for synchronizing the integrated transmission of real-time voice and text by generating text from the recognition of the voice. Our calculations and results proved that text data for real-time transmission can very efficiently be produced in this way with a negligible delay compared to the other conventional methods. There are different factors such as faster speaking rate, better speech recognition tools and algorithms and better environmental conditions which may vary the requirement and the size of the buffer used for synchronization. We have calculated the delay in speech-to-text conversion using a short list of isolated words and we intend to enhance this work for a large vocabulary continuous speech recognition system that would reflect more generic speech recognition.

REFERENCES

[1] "RFC 2793 RTP Payload for Text Conversation", www.faqs.org, Last accessed on December, 2004.

[2] "Speech Recognition", http://en.wikipedia.org/wiki/Speech_ recognition, Last accessed on January, 2007.

[3] I. Lee Hetherington, Han Shu, and James R. Glass, "Flexible multi-stream framework for speech recognition using multi-tape finite-state transducers", in support IEEE.

[4] Hugo Meinedo and Joao P. Neto, "Combination of acoustic models in continuous speech recognition hybrid systems", *in Proceedings ICSLP 2000*, Beijing, China, 2000.

[5] Claudio Becchetti, Lucio Prina Ricotti, "Speech Recognition: Theory and C++ Implementation", John Wiley & Sons Inc., 2000.

[6] "Using WAV File Input with SR", http://msdn. microsoft.com, Last accessed on January, 2007.

[7] "RFC 1889-A Transport Protocol for Real-Time Applications", www.faqs.org, Last accessed on December, 2004.

[8] John Harrington, "Interactive Visual Basic 5 Interactive Courses", Techmedia, 1998.

[9] "Automatic Speech Recognition and Understanding", www.ewh.ieee.org, Last accessed on October, 2004.

[10] "Queuing Theory", http://www.uni-koblenz.de/~kgt/Learn/Textbook/node206.html, Last accessed on November, 2004.

[11] "Dragon Naturally Speaking", www.transcriptiongear.com, Last accessed on November, 2004.